Document No. 1835-001
12 July 1990

SGML Lessons Learned for the Software Technology for Adaptable, Reliable Systems (STARS) Program

Contract No. F19628-88-D-0032

Task IR65-SGML Document Descriptions

CDRL Sequence No. 1835-001

12 July 1990

Prepared for:

Electronic Systems Division
Air Force Systems Command, USAF
Hanscom AFB, MA 01731-5000

Prepared by:

IBM Federal Sector Division
800 North Frederick Avenue
Gaithersburg, MD 20879

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	July 12, 1990	Final

4. TITLE AND SUBTITLE
SGML Lessons Learned

5. FUNDING NUMBERS
C: F19628-88-D-0032

6. AUTHOR(S)
S. Kutoroff

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
IBM Federal Sector Division
800 N. Frederick Avenue
Gaithersburg, MD 20879

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Electronic Systems Division
Air Force Systems Command, USAF
Hanscom AFB, MA 01731-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
CDRL Sequence No. 1835-001

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

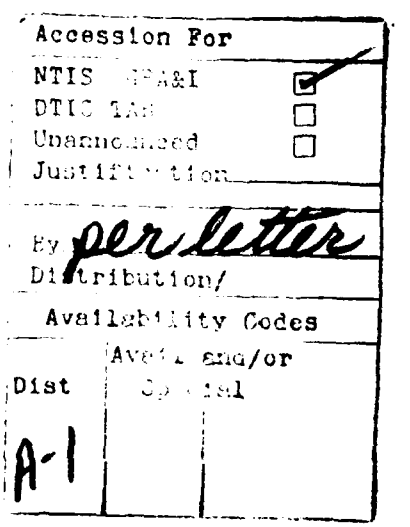
SGML (Standard Generalized Markup Language) is an international standard for representing the elements and structure of electronically stored text. SGML uses Document Type Definitions (DTD's) to unify the structure of various kinds of documents.

This document summarizes the experience of building a new DTD

14. SUBJECT TERMS	15. NUMBER OF PAGES
STARS, SGML, Standard Generalized Markup Language	19
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

TABLE OF CONTENTS



1. Introduction

Task IR65 CDRL 1810, DTD Creation, documented the foundation for using the Standard Generalized Markup Language (SGML, ISO 8879) within the STARS program. That document described how SGML differs from other markup systems and why SGML is the appropriate tool for documentation in the STARS program.

CDRL 1820, Formatting Recommendations, introduced the proposed STARS document DTD for technical reports. It included a new DTD and formatter for briefings and it updated the STARS Ada software for producing technical reports using the Report DTD. The briefing DTD and formatter provide an SGML application for preparing landscape presentation overhead transparencies. The report DTD and formatter updates added several new features to improve document appearance. These features were:

- o added the <cmptr> element for text to be presented in a `computer` style,
- o added differentiation between <graphic>, <chart>, and <figure> elements,
- o changed the DTD to allow additional end tag omissions, and
- o correction of several bugs in document appearance.

The proposed STARS document DTD was introduced in CDRL 1820. The DTD was derived from the latest available release of MIL-M-28001. Most changes involved removal of markup items not needed in the STARS program. Other changes were at the suggestion of Unisys and involved some enhancements to the 28001 DTD, such as support for 4 additional levels of subparagraphs. Since it is relatively complex, the STARS document DTD presents problems for users in learning the markup and applying it to their documents. Since its release the STARS document DTD has been used in the preparation of product reviews for CDRL 1800, SGML Product Review. The STARS document DTD was used with Software Exoterica's CheckMark product on an Apple Macintosh IIcx computer. Several problems were identified and corrective actions taken.

CDRL 1800, SGML Product Review, described a number of commercial SGML products that are applicable to the many phases of document publishing, distribution, storage, and revision. The products described are available for a wide variety of personal computers, workstations, mini-computers, and mainframe systems. In today's SGML market place there is no shortage of products; but there are limitations on which product may be used with a particular platfon.. and within a given budget.

This document will discuss the lessons learned in the performance of this task as well as describe the options available to SGML users and the impact of SGML on STARS contractors and subcontractors. Recommendations will be made on SGML products for subcontractors to use in document preparation as well as on support issues within the STARS program.

SGML software products fall into several categories (examples are in parenthesis):

1) DTD based SGML editors (XGML CheckMark, TextWrite, Author/Editor),
2) parsers (XGML Validator, ParseStation),
3) formatting systems (DL Pager, SoftQuad Publishing System),
4) SGML Publishing Systems (Compugraphic's Automated Publishing System), and
5) conversion products (FastTag, XGML Translator).

Each of these product categories has a place in the STARS program in support of document creation, publishing, review, and distribution. The SGML marketplace has matured and is becoming competitive as a wider variety of products are offered from a growing number of

vendors.

2. Lessons Learned

The preparation of documents using SGML starts with the creation of document content and ends with delivery of completed products. This section will discuss the impact of SGML on each phase of document production and provide evaluations of some commercial SGML software.

2.1 SGML Syntax Directed Editing

Computer documentation is often prepared with a WYSIWYG (what you see is what you get) word processor on a personal computer. These systems have the advantage of low price, ease of use, and the ability to produce high quality output. In comparison, editing a document structured with SGML markup is a tedious and error prone process. The author must be concerned with element tagging and the conventions of the DTD in use. Changes which are made to the document file must be verified against the document structure defined in the DTD before the format process. Even with a fast SGML parser, this process is comparable to editing and compiling a program before linking and executing it. From a user perspective, SGML has made a simple task complex.

Syntax directed editing makes editing a program more efficient and offers the same benefit to document preparation with SGML as it does with creating computer software. The current crop of DTD based SGML editors takes syntax directed editing one step further than commonly found in program editing. DTD based SGML editors ensure that the document will parse through an SGML parser (analogous to compilation) whereas most syntax directed programming editors cannot assure compilation of the source code. For this reason the use of DTD based SGML editors is highly recommended to reduce training costs and simplify SGML document production.

DTD based SGML editors are offered for a number of platforms and these products are described in IR65 CDRL 1800. Most of these products are sold as two separate programs, an editor and a program to translate a DTD into a form used by the editor (CheckMark from Software Exoterica is an exception). For those systems which are sold in two parts, the editor is usually priced much lower than the tool which allows use of new DTDs. The assumption is that DTD development will be done by few and the DTD products will be used by many in preparing documents.

Contractors and subcontractors need only buy multiple copies of the editing component of these DTD based SGML editors and one copy of the DTD preparation tool. The editors are sold in the same price range as high-end word processors. They do require more memory and faster computers than would be needed for a word processor, TextWrite from IBM requires a PS/2 computer under OS/2. Some of these editors are available with a CALS option; the STARS document DTD is a modified subset of the CALS DTD. A document prepared to the STARS document DTD will require some editing to be processed with CALS software. If an author uses the additional levels of sub-paragraph nesting allowed in the STARS document DTD, the document will not process with the 28001 DTD.

2.1.1 Commercial DTD Based SGML Editors

Using DTD based SGML editors will permit contractors and subcontractors to prepare valid SGML documents for delivery to STARS without access to any other tools. The formatting and printing of these documents will require other expensive tools, but these tools are not needed by each contractor and subcontractor. The formatting tools need only be provided at a central

site, such as the STARS Technology Center. The following two DTD based SGML editors were evaluated.

2.1.1.1 Software Exoterica's CheckMark

CheckMark was found to be a powerful SGML text editor. CheckMark acts like a character based text editor with SGML features layered upon it. It supports all SGML language features, with the exception of Link features. A document prepared with minimization or short references will be saved exactly as entered and the saved file is formatted as a text file that can be processed by another application or transmitted to another computer. CheckMark can also be used to normalize a document. The CheckMark parser operates in background and will continue document parsing after errors have been detected. SGML parsing can easily be disabled if needed.

CheckMark lacks graphic on screen renditions of the tags but instead provides a powerful system for preparing SGML documents imported from other sources and for validating documents while they are being edited. The document window has a validation bar across the top which shows the relative position of errors in the document. A mouse click over an error bar brings the display window to the position of the error and shows the error message in another window. The validation bar also shows those areas of the document which are not parsed due to edits and disabling the validation feature. Since the checking works in background, it does not interfere with the editing process in any way. CheckMark has no facility for formatted printing and can handle graphics by reference only.

CheckMark requires a minimum of 2.4 megabytes of application memory on an Apple Macintosh; however, it may need more memory depending on the complexity and size of the DTD and the document. CheckMark includes a powerful 'grep' search and replace capability in addition to the simple text searches expected with an editor. Checkmark will allow editing but not parsing if sufficient memory is not available, it warns that the XGML engine could not be started in this case. CheckMark costs $495 and is supplied with a small manual that does little to explain its operation.

2.1.1.2 SoftQuad's Author/Editor

Author/Editor from SoftQuad is an example of an editor that requires a DTD to be preprocessed prior to use with the editor. Author/Editor has a unique on screen graphic tag rendition, it permits checking to be disabled, and it allows the assignment of formatting information to a tag as a way to enhance the on screen display and permit printing. The formatting capability makes the program visually appealing and permits limited WYSIWYG style editing. Author/Editor is available on the Apple Macintosh and Sun workstations. Author/Editor permits document specific entity definitions to be defined and used. Author/Editor costs $495 for an Apple Macintosh and starts at $995 for a Sun Workstation. The user manual is comprehensive and provides ample descriptions of its features and a decent introduction to SGML.

Author/Editor saves its documents in a special format; however, an export command is available for saving in text only form for export to another program or computer. The program acts as a normalizer for exported files, i.e. exported documents contain all open and close markup tags.

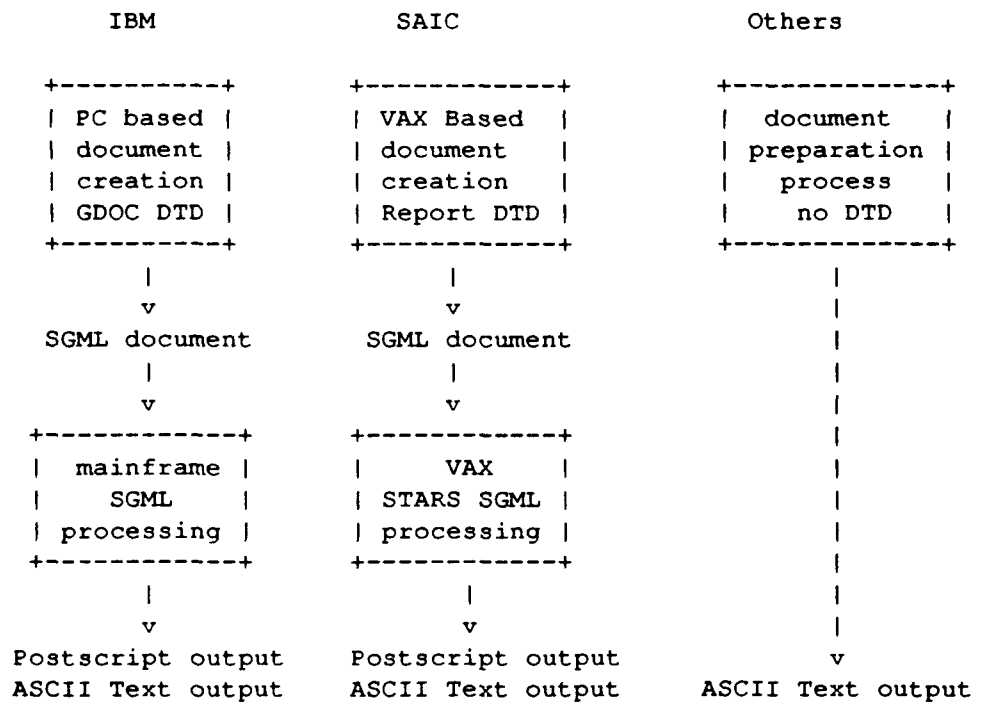
Author/Editor is provided with prepared DTDs from the American Association of Publishers and a condensed CALS DTD. Listings of these DTDs are not provided and cannot be

obtained from Author/Editor.

2.2 SGML Formatting Issues

As noted in IR65 CDRL 1810 there is no standard to specify the formatting to be used for documents prepared using SGML. Standards development is a slow process which requires consensus among representatives of companies with diverse interests in the outcome. In the absence of standards vendors have provided workaround tools to interface SGML to alternative formatting applications, often using a translation process to another markup language.

IBM has delivered its Q and R increment documents in SGML using a DTD called GDOC and processing its documents on a mainframe computer using SGML Translator Document Composition Facility (DCF) Edition. IBM subcontractor SAIC delivered its Q and R increment documents in SGML using a DTD called Report. SAIC used the STARS developed SGML software which does not fully implement ISO 8879 and permits markup which is not accepted by the commercial products. SGML documents that are produced by these two systems have completely different appearances and are incompatible. Other contractors use document preparation systems not based on SGML and deliver documents as text files with varying styles and paginations. This process is illustrated below:



Both the DCF and the STARS SGML software permit the same source document to be processed to either ASCII text files or to the Page Description Language (PDL) Postscript, a product of Adobe Systems. The text files are useful for on-line reading. Postscript permits high quality laser printing on printers which support the Postscript language.

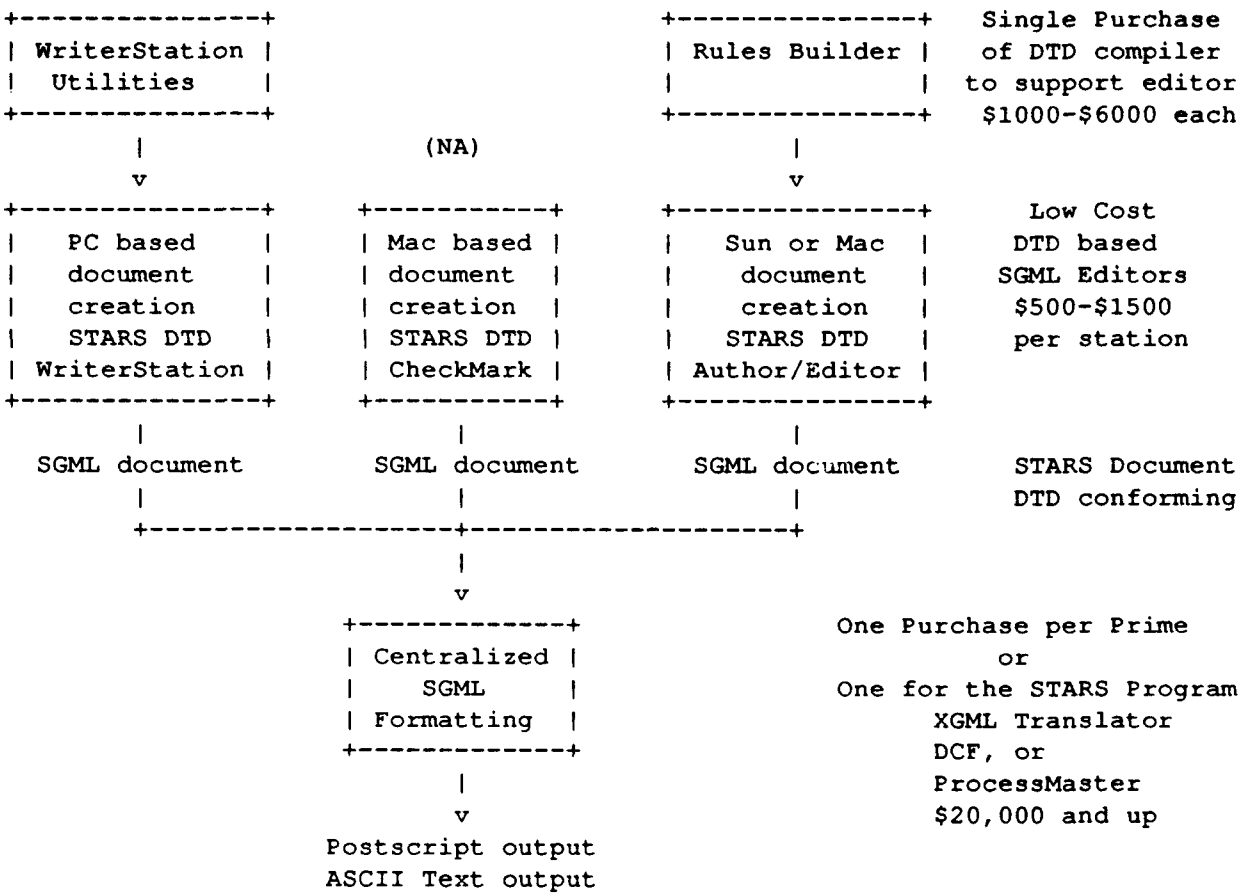
When the STARS document DTD is adopted IBM will require a new DCF formatting specification to produce printed output for documents using the STARS document DTD; development of this specification has begun. SAIC will not be able to produce printed output using the STARS SGML software since the STARS document DTD exceeds the capabilities of

the STARS SGML parser. Instead SAIC will be able to produce valid SGML documents using DataLogics ParseStation on the STARS VAX computer or by using the software purchased for evaluation under IR65. This software includes a parser that runs on a 80386 based PC computer with at least four megabytes of memory (XGML Normalizer), a DTD based SGML editor that has limited formatting ability (Author/Editor), and a DTD based SGML editor that offers no formatting (XGML CheckMark).

Unisys has used Author/Editor on a Macintosh and has indicated interest in the Sun version recently announced by SoftQuad. Unisys delivered several documents with SGML Report markup in the Q increment. Unisys has also been incorporating limited SGML program markup in their Ada code deliveries. No product deliveries from Boeing have been observed to use SGML in any form as of June 1990; however, two documents from the Q increment were located which describe SGML activities, task BQ12 CDRL 320 and task BQ12 CDRL 360.

Since there will be only one DTD in use for technical reports, only one format specification needs to be prepared if all formatting is done at a central site. Since there is no standard specification language for a formatter, each system will require application specific development. As long as authors have access to DTD based SGML editors, they will not be impacted as document delivery will be validated to the STARS document DTD.

A central site could offer formatting and printing services and support STARS contractors lacking a local capability. The STARS Repository computer systems are situated to offer such services to all STARS contractors and therefore promote increased utilization of STARS resources. Network access to the central site will allow remote users to use the File Transfer Protocol (FTP) to exchange files with their local computer. This will be possible once the repository computer is on the Internet. The suggested process is shown in the following figure:



Using this multi-level approach minimizes the cost for the contractors and subcontractors but puts an additional burden on the repository computer or another central computing facility. Alternatively, each prime could purchase their own formatting application software and deliver all final document forms (SGML, Postscript, and ASCII Text). The investment for the smaller subcontractors is the price of a DTD based SGML editor, which start at about $500 and can range up to $5,000 depending on the selected platform and additional capabilities purchased.

One solution to the formatting problem is to use a program to translate from SGML to a common document processing language, such as LATEX. XGML Translator from Software Exoterica has been evaluated and found to effectively perform translations and SGML parsing. A translator was defined in one day for the Report DTD to LATEX and used for printing documents on the STARS VAX system using a demonstration copy of XGML Translator.

NOTE: XGML Translator located a problem with the Report DTD that had been suspected based on problems in coding its Text Composition Specification. The text for a heading could not be formatted for output or used in the table of contents due to the way it was defined. A modification to the DTD was made which requires no changes to most documents already marked up using the DTD. The content model of the <head> tag was re-defined as:

```
<!ELEMENT head      - o ( headtitl , ( %subbody; )* ) >
<!ELEMENT headtitl o o ( %text; ) >
```

```
Previously, the content model was:

<!ELEMENT head      - o ( %text; , ( %subbody; )* ) >
```

The new element `headtitl` is required, but its start and end tags may be omitted. In this way, existing documents continue to parse without change.

2.3 SGML Implementation Problems

It is important to note that not all SGML products are alike. In the Ada world all compilers that purport to be Ada have passed a minimal set of tests which ensure that the product does what is claimed. An SGML conformance standard is being progressed through the American National Standards Institute committee X3V1. Until conformance issues are resolved there will be problems with interpreting the standard. In addition many published books and articles on SGML contain examples which demonstrate misleading use of SGML.

2.3.1 Entities

The use of SGML entities to shorten standard text references is often shown as one advantage of SGML. Entities are predefined text macros that may be included by reference, for example `&stars;` may be defined to expand to "Software Technology for Adaptable Reliable Systems" by the entity declaration:

```
<!entity stars "Software Technology for Adaptable Reliable Systems">
```

In most implementations of SGML parsers entities may not be defined in a document instance, they may only be defined in the DTD. The consequence of this limitation is that authors may not declare entities unique to the documents they create and they will be limited to entities declared in the DTD.

Using marked sections is a case where the examples in the annex to the SGML standard are misleading (the annexes do not form an integral part of the standard). Marked sections allow a user to have portions of a document included in parsing or ignored by the parser. This feature permits one source document to be used in the production of multiple versions of a final document. The annex to the standard shows an example where a single entity definition may be used to control document production using marked sections. The following is extracted from annex B.8 of ISO 8879:1986 on page 83:

```
<!ENTITY % systema "IGNORE" >
<!ENTITY % systemb "INCLUDE" >
```

```
The "IGNORE" keyword would not be used directly in any marked section
declaration.  Instead there would be a reference to one of the two
system-dependent entities.  Given the previous declarations, the instruction
for "System A" in the following example would be ignored, while the one
for "System B" would be executed."
```

```
<![%systema;[<?instructions for System A>]]>
<![%systemb;[<?instructions for System B>]]>
```

This and the other examples show cases where document specific entities exist as the "systema" and "systemb" version of a document instance is not likely to be known to the DTD designer. In most implementations these entities must be defined in the DTD. The STARS SGML parser, the Sobemap parser (as used in Mark-It), and the parser used in Author/Editor are known to accept entity declarations in a document instance.

There is one way to add such document specific entities to a DTD by using an external reference to the DTD content, as shown below:

Source Document Instance:

```
<!doctype doc system "sgml:report.dtd" [
   <!entity mo  cdata "<">
   <!entity mc  cdata ">">
   <!entity me  cdata "</">
]>
<doc>
... document content ...
</doc>
```

This feature can be used only if the external file which contains the DTD, "sgml:report.dtd", omits the DTD doctype declaration and the delimiters associated with it. The Report DTD is shown below as it would be stored in the file "sgml:report.dtd" to be used in this fashion:

```
<!ENTITY  % misc    "bullet | seqlist | chart | figure | graphic | note" >
<!ENTITY  % subbody "head | para | %misc; " >
<!ENTITY  % text    "( #PCDATA | ( indxflag | cmptr ) )*" >

<!ELEMENT report    - - ( front? , bodym , rear? ) >
<!ELEMENT front     - o ( titlepg , contents? , iluslist? , deflist? ) >
<!ELEMENT titlepg   - o ( title | docno | date |
                          reldate | author | address )* >
<!ELEMENT indxflag - -   RCDATA  >
<!ELEMENT cmptr     - -   RCDATA  >
<!ELEMENT title     - o ( #PCDATA ) >
<!ELEMENT docno     - o ( #PCDATA ) >
<!ELEMENT date      - o   EMPTY >
```

```
<!ELEMENT reldate  - o ( #PCDATA ) >
<!ELEMENT author   - o ( #PCDATA ) >
<!ELEMENT address  - o ( #PCDATA ) >
<!ELEMENT contents - o   EMPTY >
<!ELEMENT iluslist - o   EMPTY >
<!ELEMENT deflist  - - ( term , def )* >
<!ELEMENT term     - o ( #PCDATA ) >
<!ELEMENT def      - o ( #PCDATA ) >
<!ELEMENT bodym    - o ( section | %subbody; )* >
<!ELEMENT section  - o ( sectitle? , ( %subbody; )* ) >
<!ELEMENT sectitle - o ( #PCDATA ) >
<!ELEMENT head     - o ( %text; , ( %subbody; )* ) >
<!ELEMENT para     - o ( %text; | %misc; )* >
<!ELEMENT bullet   - - ( item+ ) >
<!ELEMENT item     - o ( %text; ) >
<!ELEMENT seqlist  - - ( item+ ) >
<!ELEMENT chart    - -   RCDATA >
<!ELEMENT figure   - -   CDATA >
<!ELEMENT graphic  - o   EMPTY >
<!ATTLIST graphic        xrefid CDATA "">
<!ELEMENT note     - o ( %text; ) >
<!ELEMENT rear     - o ( appendix* , deflist? , index? ) >
<!ELEMENT appendix - - ( apdxtitl , ( %subbody; )* ) >
<!ELEMENT apdxtitl - - ( #PCDATA ) >
<!ELEMENT index    - o   EMPTY >
```

Storing a DTD in this way allows it to be extended with entity declarations as required by the document.

2.3.2 Character Data

SGML allows the content of an element to be declared as one of several possible content models:

- o Parsed Character Data (PCDATA),
- o Replaceable Character Data (RCDATA),
- o Character Data (CDATA),
- o Non-SGML Data (NDATA),
- o Specific Character Data (SDATA),
- o Processing Instruction (PI), or
- o No content (EMPTY).

Each of these content models has a unique meaning and is appropriate in some application. PCDATA is used when the content of an element may include additional markup which must be parsed. RCDATA is used when the content of an element may include entity references but will not and should not include additional elements of markup (tags). CDATA is defined as content where neither entities nor markup are recognized.

CDATA has obvious application to content which could contain text that includes character sequences resembling markup, such as an end tag open (ETAGO) sequence, typically "</". Such a sequence would be included if SGML fragments were to be used in a document, as would occur if SGML were documented in SGML. One might expect that an element defined as

CDATA would be ended only by the occurrence of an end tag for that element. Such an interpretation is in conflict with markup minimization and other SGML features.

The problem with CDATA is illustrated by the STARS document DTD and CDRL 1820. The content defined for the <verbatim> tag in the STARS document DTD is CDATA (Character Data). ISO 8879 defines Character Data on page 6 as "Zero or more characters that occur in a context in which no markup is recognized, other than the delimiters that end the character data. Such characters are classified as data characters because they were declared to be so." Annex B of 8879 (not a part of the standard) states as explanation "If an element contains declared character data, it cannot contain anything else. The markup parser scans it only to locate an ETAGO or NET; other markup is ignored. Only the correct end-tag (or that of an element in which this element is nested) is recognized."

Annexes are not a part of the standard and the requirement in the last sentence above was deemed to be too great an imposition on implementers and it was officially dropped from the standard, hence the conflict with the definition on page 6. As a consequence it becomes difficult to document SGML in SGML unless alternate concrete syntaxes are used or SGML content expressed in CDATA is edited to not contain ETAGO (end-tag open) or NET (null end-tag) sequences. The result is that content which may contain ETAGO or NET sequences are best included as external SGML entity references to prevent unexpected parsing problems.

2.3.3 Mixed Content

Another problem has been identified while using the STARS document DTD on the sample file provided in CDRL 1820. There are a number of cases in the DTD where CheckMark (but not ParseStation) reported the following warnings:

```
Note on line 177:
An Element with mixed content does not permit data characters
everywhere.
Spaces and line breaks in Element 'NOTICE' may be treated as data
characters, forcing insertion of markup.
```

This error was also reported for elements INTERNATLSTD, ITEM, DEF, ENTRY, and FTNOTE. The consequence is that RE characters, record-end (carriage returns) are not permitted in some situations, commonly between certain end tags and a following start tag. The Datalogics parser does recognize these errors when parsing a document instance using the effected tags.

> NOTE: These errors also exist in the version of MIL-M-28001 used as a baseline for developing the STARS document DTD. It is expected that MIL-M-28001 will undergo further revisions and that some corresponding changes will be needed to the STARS document DTD. This is the same error detected by XGML Translator in use of the Report DTD element <head> . This error could cause problems in formatting applications.

This error was unexpected, but it is a correct interpretation of the standard. This error condition is caused by the insertion of blank lines after end-tags to visually separate sections of a document in text editing. The problem is also caused by starting a new tag on a new line after a close tag on a previous line. If the DTD does not permit content at those points in a document, the white space becomes illegal.

This is a problem in working with source documents using SGML and it can be likened to a compiler not permitting a blank line between the end statement of a procedure or a block and the

next procedure or block. This limitation is an issue only for those who must work with an SGML document. Using DTD based SGML editors does not eliminate this problem. The following shows improper SGML markup for the Report DTD:

```
<para>Paragraph text...
</head>
<head>This is a new heading...
<para>Another paragraph...
```

The above markup sequence is very common in files prepared using the STARS SGML software and the Report DTD. It has also been noted in the MIL-M-28001 DTD, and therefore the STARS document DTD. Yet a validated SGML parser will reject it due to the RE character between the close of one `head` and the open of another. This content is not proper use of SGML, but the correct form of the markup is difficult to read. This class of problem places an additional burden on the DTD designer who must consider parsing details in document design.

2.3.4 Issues with the Datalogics SGML Parser

The Datalogics SGML parser product ParseStation was purchased for the STARS computer using IR10 funds in January 1990. The Datalogics product correctly processes SGML except as noted below. ParseStation also executes very fast. The product performs as advertised; however, several problems have been identified which Datalogics has been very slow to address.

ParseStation consists of three main tools, a SGML declaration parser (SP), a DTD parser (DP), and a SGML document instance parser (IP). PARSE, a menu interface to these tools is also provided. The applications use a job file to tie a declaration, a DTD, and a document together. The Datalogics parser will not accept an SGML document which includes a document type declaration subset or a DTD combined with a source instance, a standards violation. The Datalogics literature states the product is in compliance with ISO 8879.

The SGML declaration parser contains minor errors which were found when preparing a SGML Declaration file. Test files were prepared using examples from the standard, from a reference book, and from XGML CheckMark. The Datalogics SGML Declaration parser does not accept parts of the syntax for character set definition involving DESCSET and BASESET. It will also not accept large capacity set declarations. SGML Declation files may be modified to omit the offending syntax; however, such a file is not proper with respect to ISO 8879.

Another problem with ParseStation is with the VAX image files provided for the command line interface. These programs work as Unix filter applications; they take filenames and switches as input and produce an output file. As with most Unix tools, these applications are designed to output usage information when the command is entered with no parameters. Our installation (for VMS 4.7) includes non-printing out of range data in the help output; causing the VT320 terminals to enter a non-responsive state requiring a full reset to correct. This condition was reported to Datalogics in March of 1990.

3. The Integrated Chameleon Architecture

Ohio State University has proposed forming a consortium of industrial members for the development of public-domain software for data-exchange based on SGML and known as the Integrated Chameleon Architecture (ICA). The software will facilitate the interchange of data prepared using systems other than SGML and is based upon the use of SGML as a common intermediate representation of content. The systems to be developed will use the X-Windows user interface and will be coded in the ANSI version of the C language for portability to any Unix platform. The consortium itself is being modeled after the MIT X-windows consortium.

The OSU proposal is to continue work started in 1985 and supported by the Applied Information Technologies Research Center (AITRC) in Columbus, Ohio. The AITRC funding ended on 30 June 1990. Funding from the consortium will be used to continue the development of tools based upon the data-exchange architecture developed for AITRC. The AITRC work has supported several organizations' needs for translation in both publishing and database applications. The article by Mamrak, et al in IEEE Transactions Vol 15 No. 9 lists several firms as using the tools developed under the AITRC funding.

OSU hopes that the funding from the consortium will permit the existing graduate students, doctoral candidates, and post-doctorate researchers to continue their work without interruption. The goals of the consortium are:

1. continue research into translation technology,
2. continue development of Chameleon tools,
3. produce commercial quality tools that offer significant productivity improvement,
4. offer tools with greater integration than available from vendors, and
5. offer tools with better technology than vendors.

These goals are based upon OSU's evaluation of existing commercial offerings with specific comparisons to Software Exoterica's XGML Translator and products from Yard Software. OSU representatives stated that the vendor products lag their technology and are in fact based upon technology developed and published by OSU.

The risk factors in supporting OSU compared to reliance on commercial products are:

1. OSU has no track record in production of commercial quality software,
2. potential risk of insufficient industrial sponsors,
3. potential risk of schedule expansion due to reduced funding, and
4. possible underestimation of work required.

OSU has estimated that twelve industrial sponsors will provide the required funding to produce results in one year. There were approximately twelve industrial representatives at the Integrated Chameleon Architecture demonstration held at OSU on 29 June 1990. Several of the corporate representatives expressed considerable interest in the program; however, many also wished their interest to be considered confidential. Many of the companies with prior commitments to the project were present.

The demonstration was performed in a highly constrained environment and did not permit hands on trial or an objective assessment of the software's rigor in operation. Questions revealed that the SGML software developed at OSU does not comply with ISO 8879 any more than does the STARS SGML Parser, although ICA is faster.

The benefits of membership in the Integrated Chameleon Architecture consortium are:

1. immediate access to the translation tools between LATEX and SCRIBE,
2. participation and influence over research directions,
3. access to alpha and beta copies of existing tools, and
4. advanced access to other tools developed using consortium funds,

The cost of membership is to be $25,000 per sponsor per year. This cost is comparable to the purchase price of commercial translation software such as XGML Translator ($15,000 - $20,000) along with some related tools.

Participation in this project might be best undertaken in the form of a breakthrough initiative. Funding this research may provide STARS advanced access to the technology and a direct source of reusable software. Development of a standard SGML parser for STARS and the translation tools might be sufficient justification for the risk. Participation will require the commitment of management funds and travel time for an engineer to participate in and monitor the progress of the project.

4. SGML Impact on Users

Thus far the SGML document markup requirement for STARS has not been enforced, due to concerns about impact on subcontractors, cost, schedule, and concern about SGML as a standard. Those who have used SGML for document production have learned that using SGML requires consideration of the entire document production process, including print technology. Using SGML does permit document interchange, but it does not automatically provide for the interchange of revisable documents. Revisable interchange requires the ability to exchange and use formatting information along with a DTD.

Fully revisable interchange will be possible when a standard for exchanging formatting information is established (the status of a formatting specification language for SGML was discussed in CDRL 1810). Until the time when formatting information can be as easily exchanged and reused as a DTD, there will be problems in SGML document interchange.

Commercial SGML products have been on the market since shortly after SGML was published as standard in 1986. IR65 CDRL 1800 lists SGML products for which information was available at the time it was prepared. That document shows that there are products now available for a wide variety of platforms and that lack of commercial products is no longer a concern. However, users will have to adjust to SGML and idiosyncracies previously described.

The cost impacts of using SGML within the STARS community includes:

1. buying SGML software products,
2. training authors to use these products,
3. training authors in SGML,
4. training authors to use the STARS document DTD,
5. using SGML compared to traditional methods, and
6. developing an infrastructure of supporting standards and tools to support using SGML.

DTD based SGML editors can now be purchased for as little as $500, depending on platform. Switching from a word processor to a DTD based SGML editor is not difficult, some products offer on screen formatting that is applied to document elements resulting in a WYSIWYG display. Authors will have to learn to concern themselves with content and not appearance as SGML permits them to abandon the pre-Gutenburg concept where a manuscript reflects artistic ability as much as literary skills. Learning SGML is not a great investment for an author and frees him from concern over the details and limitations of printing.

SGML validators can be purchased for $200, depending on the platform. The training and operating costs for SGML can be reduced by reliance on DTD based SGML editors to do the document validation and reduce the time and effort needed to learn the tag set for a given DTD. Such editors range in price from $500 and up. If a single editor is standardized for each platform in the program there will be a greater pool of knowledge to draw upon in working with the available tools. Such standardization will also permit the preparation and distribution of skeleton documents to be used as a starting point for authors.

Tools to prepare printed documents using SGML can be purchased for $20,000 and up; however, few of these expensive tools are needed. As noted earlier, only one site must have such a tool. In addition, projects such as the Integrated Chameleon Architecture may provide public-domain SGML software for conversion from one markup language to another.

Conversion between markup systems will always be required.

Training costs for the STARS document DTD are difficult to predict, but the cost should not be excessive since much of the STARS document DTD is not relevant to the type of documents that have been delivered to date. Tools such as Author/Editor almost eliminate the need to refer to a DTD at all and they compensate for the need to conform to a rigid document structure. As noted above, providing a skeleton document with STARS document markup will help new users get past the initial writer's block.

In the future additional standards will be required as graphics, figures, and tables become a routine part of STARS documents. Tools to convert between graphics standards and vendor formats will be as important as tools to convert from other markup systems to SGML. It is important that the STARS program look into these standards and products and become a testing ground just as for SGML itself.

There are also advantages to be gained by assigning a single organization responsibility for support of the selected tools and tracking standardization in the industry. Limiting the number of tools that require support and concentrating the expertise on those tools to a single organization should keep costs manageable.

APPENDIX A.

APPENDIX: Bibliography

Coombs, J.H., Renear, A.H., DeRose, S.J. "Markup Systems and the Future of Scholarly Text Processing", Communications of the Association for Computing Machinery, Volume 30 Number 11, November 1987, pp 933- 947.

International Organization for Standards. Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML), ISO 8879:1986, Switzerland.

International Organization for Standards. Information Processing - Text and Office Systems - Operational Model for Text Descriptions and Processing Languages, ISO/IEC JTC1/SC18/WG9 N807, 17 March 1989.

Mamrak, S.A., Kaelbling, M.J., Nicholas, C.K., Share, M., "Chameleon: A System for Solving the Data-Translation Problem", IEEE Transactions on Software Engineering, Volume 15 Number 9, September 1989, pp 1090-1108.

Shiff, B., Sharpe, P., Spencer, R., "Author/Editor User's Manual", SoftQuad, 1989.

Science Applications International Corporation. REPORT.DTD, prepared under contract N00014-87-C-2386 to the Naval Research Laboratories for the STARS Foundation Project, 1988.

Software Exoterica Corporation. XGML Translator(tm) XTRAN Programmer's Manual, Version 1.0.

MIL-M-28001A (DRAFT COPY), as provided for review by Electronic Publishing Committee, Computer-Aided Acquisition and Logistics Support (CALS).